LA-UR- 04-1114

Approved for Public Release
Distribution is Unlimited

Title

Blue Gene: A Performance and Scalability Report at the 512-Processor Milestone

Authors

Kei Davis, Adolfy Hoisie, Greg Johnson, Darren Kerbyson, Mike Lang, Scott Pakin, Fabrizio Petrini

Published

PREPRINT

Modeling, Algorithms, and Informatics Group

Performance and Architecture Lab

http://www.c3.lanl.gov/par_arch

Blue Gene: A Performance and Scalability Report at the 512-Processor Milestone

*Kei Davis, Adolfy Hoisie, Greg Johnson, Darren Kerbyson, Mike Lang,
Scott Pakin, Fabrizio Petrini*

Performance and Architecture Laboratory (PAL)
Computer and Computational Science Division (CCS)
Los Alamos National Laboratory

1. Summary

This set of measurements was done on the 512-processor Blue Gene machine at IBM Watson. The analysis based on the measurements deals with the overall performance of the machine from low-level characteristics to applications.

We have compared the measured values for the benchmarks in our suite against modeled predicted numbers. In general, the error bars were relatively low. We also made predictions using our models for the performance of two important applications from the ASCI workload on the full BG/L configuration of 64,000 processors.

Given the "milestone" character of these benchmarks, extra precautions have been used in the protocol of the runs. For example, a number of possibilities related to the usage mode of the co-processor were considered: heater, co-processor and virtual modes.
The torus was not available, instead the topology was a mesh.

The report is structured as follows. Section 2 gives an architectural description of the machine. Section 3 describes the communication characteristics of the networks. Section 3 analyzes the issue of "computational noise". Section 4 deals with single processor performance issues. Section 5 concerns itself with application performance and scalability, including performance prediction. Section 6 compares the predicted performance of BG/L against the performance of ASCI Q. We summarize in section 7.

2. Architectural Description

The basic building block in the system configuration is a board consisting of 32 processors. Each processor has 256 Mbytes of memory and no local disk**.** The processor is a dual core embedded version of the Power-PC 440. Each core has a 2.8 GF/s peak performance. The system on a chip design incorporates dual CPUs, and their 3 levels of cache, and 3 interconnects (GigE, JTAG, and Torus). The caches are coherent between the two cores on a board. The sizes of the cache are 32KB L1 data cache, and a 4MB L3 cache. There is also a 2KB prefetch buffer serving the role of a very small L2 cache.

There are 4 types of processor configurations for the dual PPC core of the processing nodes:

- Heater mode – one processor run the other is idle,
- Communication mode – one processor is dedicated to communication and the other for general processing.
- Symmetric mode – both CPUs process and communicate.
- Virtual mode –two threads per CPU can run.

The target processor frequency is 700 MHz. The packaging is very dense, allowing for a standard size cabinet to hold 1024 processors.
The topology of the communication network is a 3-D torus of size 32 by 32 by 64 for the full 64K-processor configuration.

The machine on which we conducted our experiments was a 512 processor midplane arranged in an 8 by 8 by 8 mesh. The processors were clocked at 500 MHz.

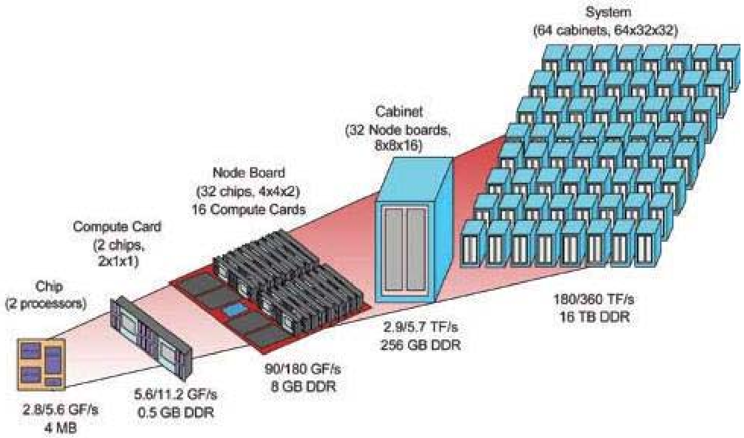

Figure 1. BG/L General Architecture (from http://www.cacr.caltech.edu/News/sc02/gallery/pages/bluegenel.htm

The midplane was constructed as follows; 2 nodes per compute card, 16 compute cards per node board, 16 node boards per 512 midplane. A node card has up to 32 compute nodes and 4 I/O nodes.

A detailed description of the Blue Gene architecture can be found in [1] and [2].

The runs are performed in "partitions" which are statically assembled based on an XML database. Each node card (32 processors) can be partitioned into as small as 8 compute nodes. Each of these partitions then forwards all I/O requests to the I/O nodes. This keeps the kernel on the compute nodes very streamlined.

3. Computational noise

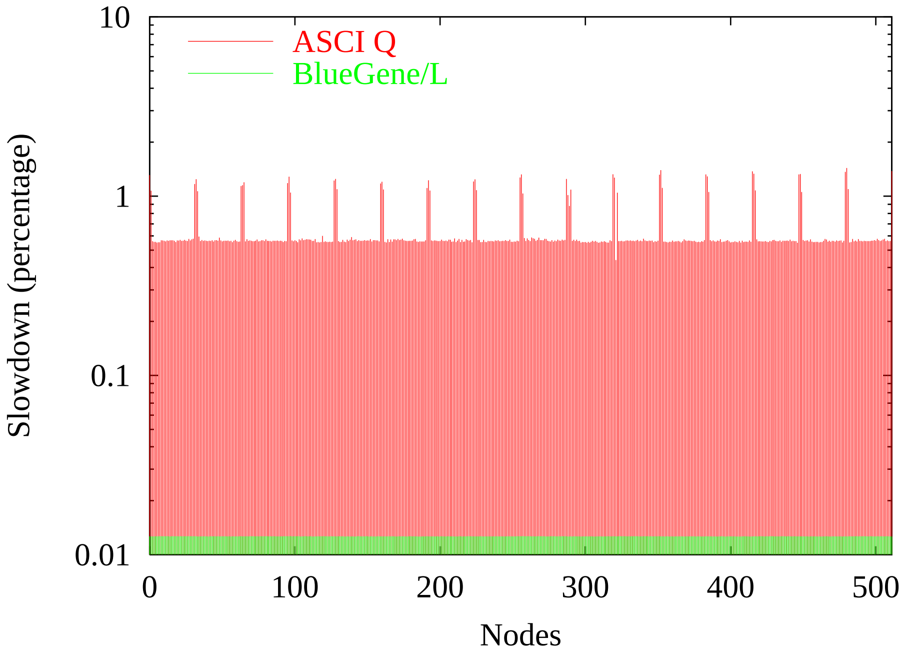

Figure 2. Slowdown due to computational noise on a per node basis.

Figure 2 above shows the slowdown due to computational noise on a per node basis. The comparison with the system noise on ASCI Q machine is also indicated in the figures. The level of intrusion of the operating system is minimal, two orders of magnitude less than traditional clusters. The bottom line is that the noise is negligible on the Blue Gene, which was expected given the micro-kernel based system software and the tight synchronization of the kernel-level activity. For more information on this methodology see [3].

4. Network Performance

The basic network performance of the 3-D mesh network is approximately 110 MB/s asymptotic bandwidth (figure 3) and 6 μs latency (figure 4). Overall the MPI implementation was stable and didn't have any performance problems.

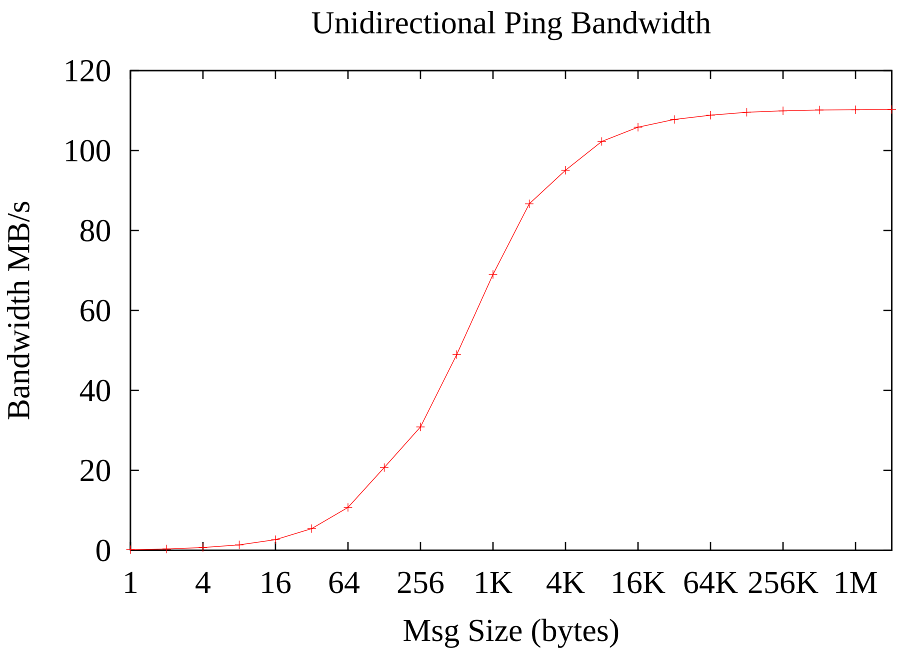

Figure 3. Unidirectional ping bandwidth.

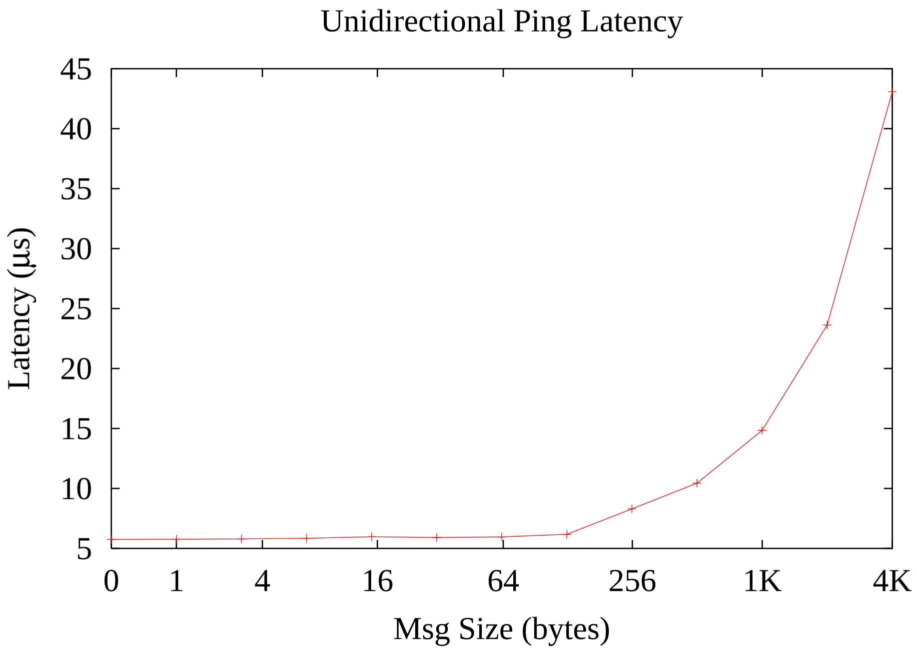

Figure 4. Unidirectional ping latency.

Figures 5 and 6 show the unidirectional bandwidth and latency seen by processor 0 when communicating to any other processor. It is remarkable to see that the system is so deterministic that it is almost possible to reverse-engineer the topology of the network. For example, from the bandwidth graph we can easily identify groups of 8 processors, that are aligned on a single row, and 8 boards of 64 processors each.

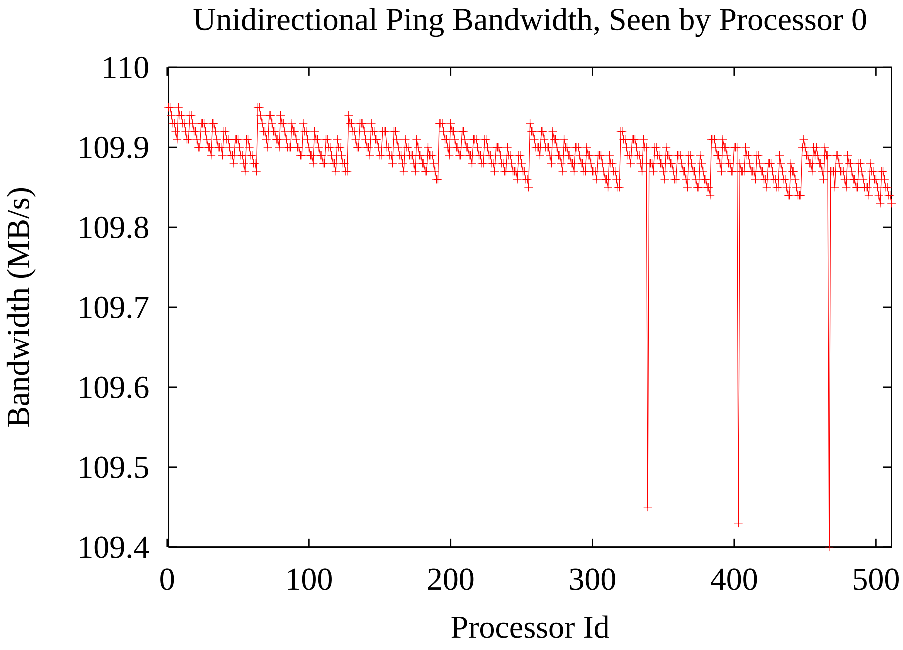

Figure 5. Unidirectional ping bandwidth seen by proc 0.

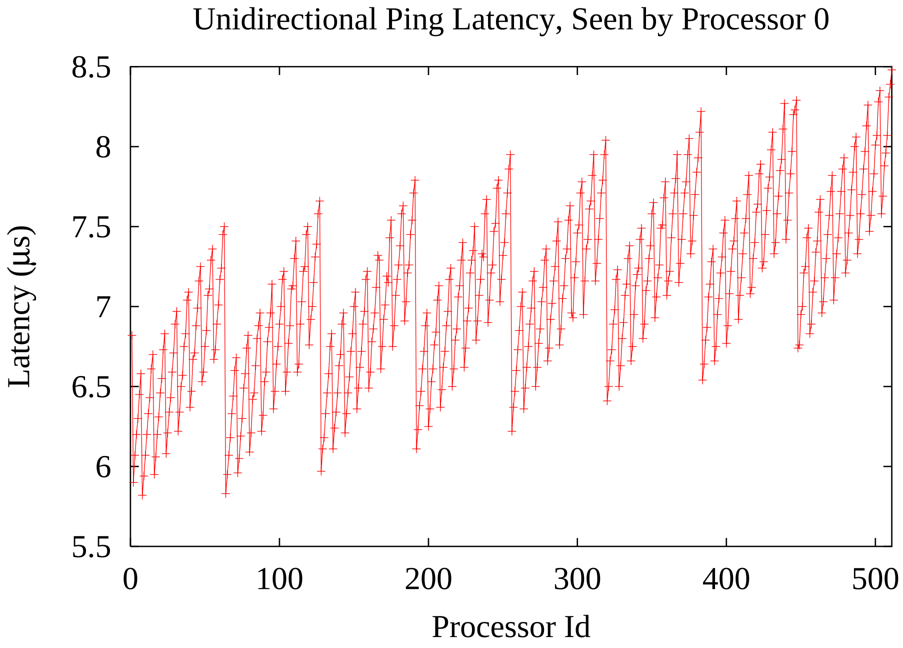

Figure 6. Unidirectional ping latency seen by processor 0.

The following two graphs (figures 7 and 8) show the network performance under bi-directional traffic: two processors, residing on two distinct nodes send messages to themselves. The performance is similar to the unidirectional case, and shows that the network and the network interface can handle bi-directional traffic without any performance degradation.

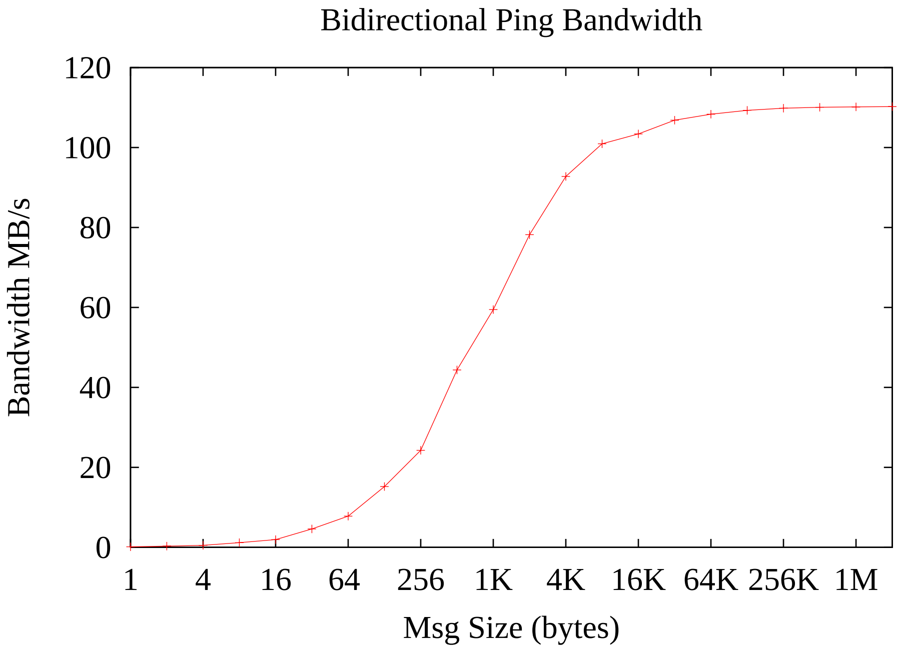

Figure 7. Bidirectional ping bandwidth.

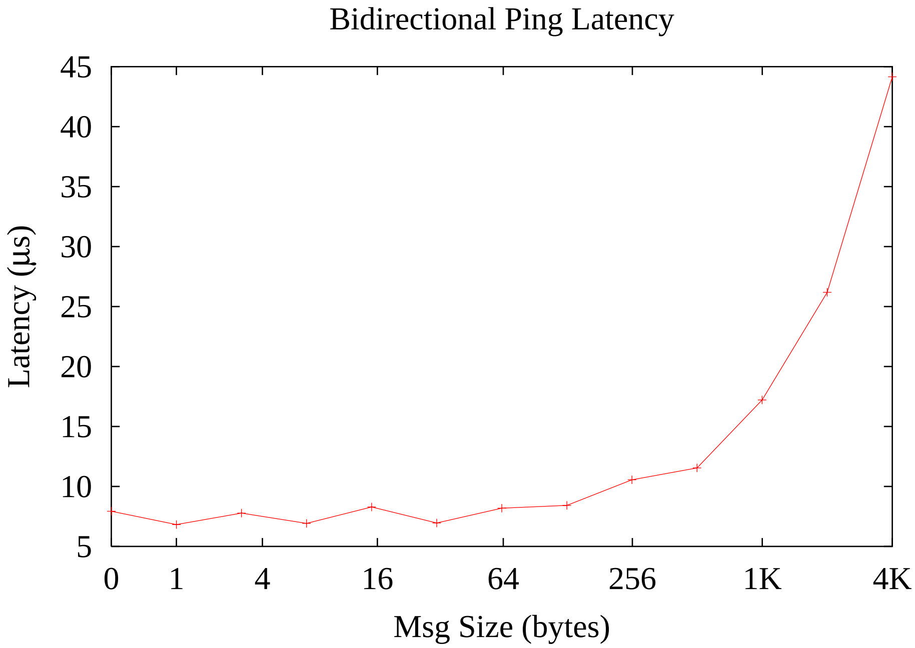

Figure 8. Bidirectional ping latency.

In the complement permutation pattern depicted in Figure 9, each processor sends messages to a partner processor that is identified by the bit complement of the bit-string representing the sender, modulo the total number of processors. The complement traffic is a good indicator of how the network as a whole behaves under heavy traffic and whether the actual bisection bandwidth that can be achieved under stress. We can see that the pairwise bandwidth changes, according to the location of partners on the topology and the congestion encountered along their communication path. This graph exposes the properties of the 3-dimensional mesh, whose performance is sensitive to process mapping.

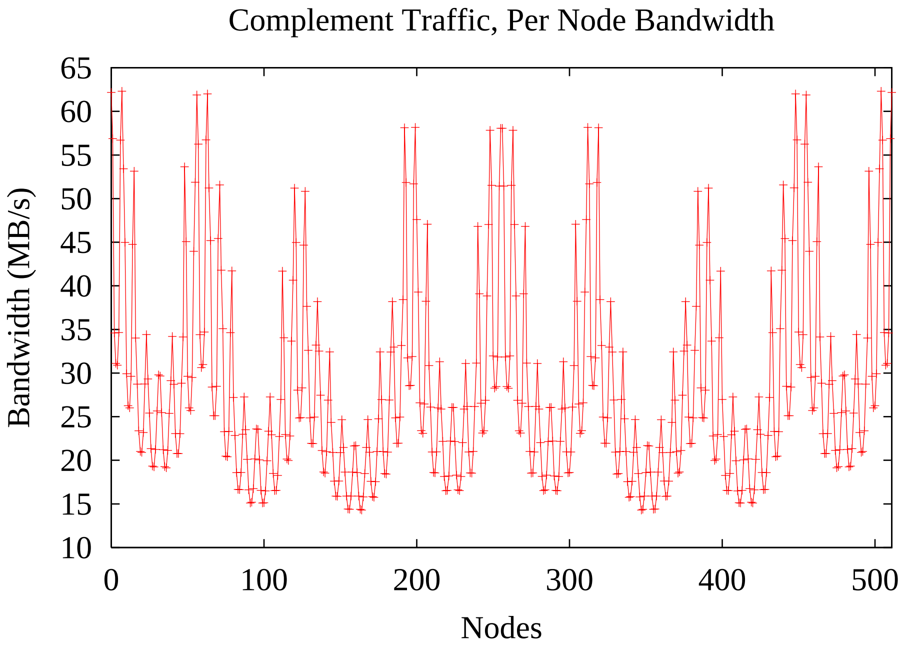

Figure 9. Complement traffic.

In the hot spot traffic, multiple nodes are sending messages to a single destination. As in the previous traffic pattern, the performance under hotspot is sensitive to the mapping of the hot node in the 3-D mesh topology. In the following graph (figure 10), the hot node is positioned in one of the corners of the 3-dimensional cube (node 0). The graph shows some interesting properties of the network. With two processors we get the basic 110 MB/sec, and the performance degrades to 92 MB/sec with 8 nodes, because they are aligned on a single row and thus the message routing experiences some degree of congestion. With more than 8 nodes the bandwidth increases because it is possible to use more than one incoming link. Nevertheless, the incoming bandwidth never reaches the peak performance of 330 MB/sec (considering the three times the incoming bandwidth of a single link). We speculate that this is related to the network interface that is not fast enough to pull messages off the network at full link speed.

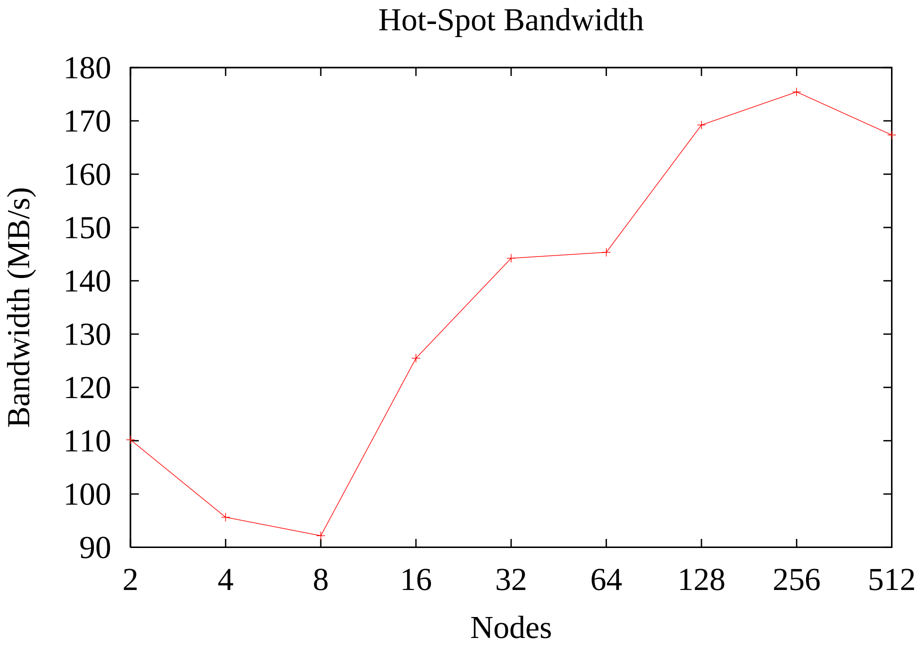

Figure 10. Hotspot bandwidth.

In the following 2 graphs we show the performance of the tree network and the barrier synchronization latency (figure 11), which is only a few microseconds and the asymptotic broadcast bandwidth (figure 12), which is insensitive to the number of nodes. In both cases the tree network delivers excellent, scalable performance.

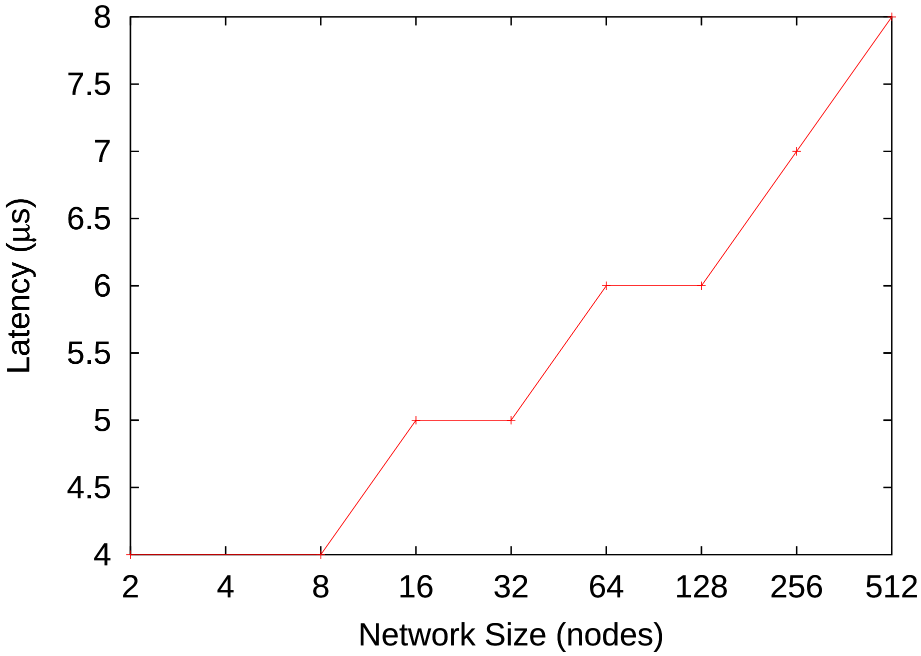

Figure 11. Barrier performance.

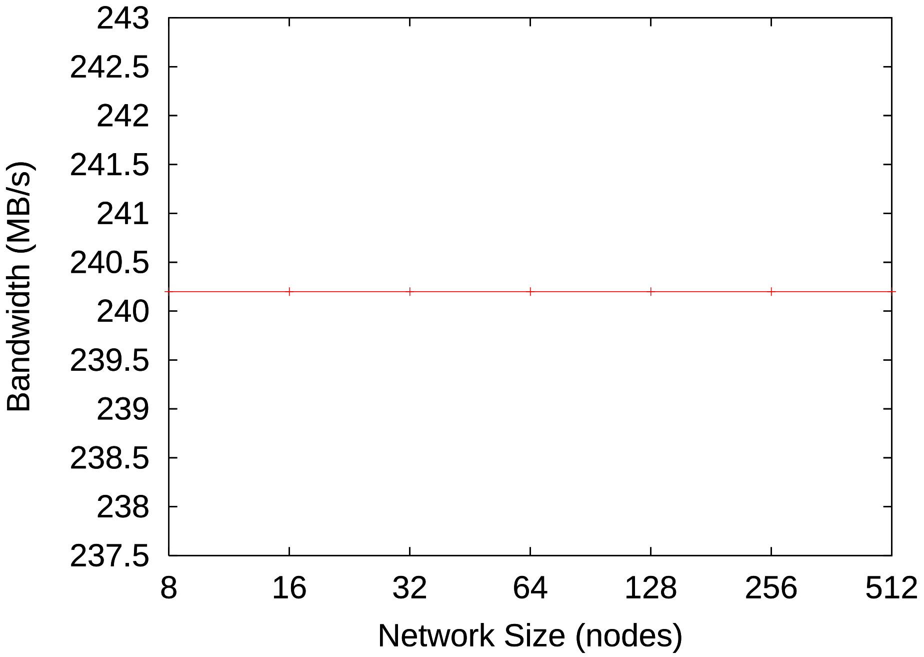

Figure 12. Broadcast bandwidth.

5. Single processor benchmarks

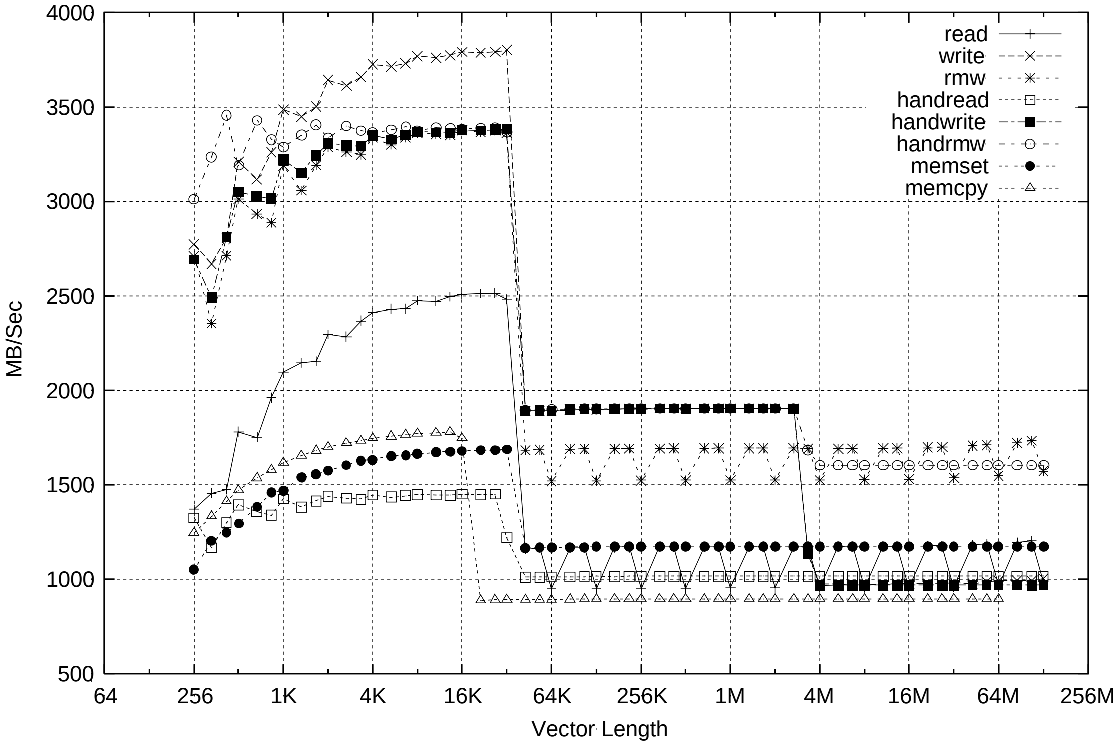

Figure 13. Cachebench on the single-processor BG/L.

The preceding figure 13 shows the results of running CacheBench [http://icl.cs.utk.edu/projects/llcbench/cachebench.html] on a single BG/L node. Each node has a 5.5 GB/s link to main memory, which establishes an upper bound on the memory bandwidth. For data that fit in cache, a BG/L node can read/modify/write up to 3,377 MB/s, which, for a 500 MHz CPU is proportionally slightly better than the peak read/modify/write bandwidth achieved, e.g., by a 2 GHz Opteron (10,176 MB/s).

Once data no longer fits in the L1 cache, the BG/L node sees some unusual bimodal behavior in memory bandwidth on the read, write, and read/modify/write tests but not on the hand-optimized (i.e., manually unrolled in groups of eight accesses) read, write, and read/modify/write tests. Performance is consistent across runs of CacheBench. The mean read/modify/write bandwidth to main memory is 1,643 MB/s, which again compares favorably to a 2 GHz Opteron (2,983 MB/s).

6. Application Performance

6.1 Measurements

Figures 14–17 show the performance and weak-scaling behavior of Sweep3D running on BG/L. We measured performance using two problem sizes and two blocking factors. For the problem sizes, we used 50×50×50 cells and 5×5×400 cells. These were each blocked using 1 k-plane/1 angle and 10 k-planes/3 angles. Each graph shows two curves: one in which the processor grid is organized with more CPUs in the *x* dimension than in the *y* dimension and one with more CPUs in the *x* dimension than in the *y* dimension. The intention of these experiments is to determine how well Sweep3D performs and how well it scales for both coarse- and fine-grained problems and with different message sizes. Also, by varying the processor grid, we can determine the importance of data placement to application performance.

The first experiment, with data presented in Figure 14, represents a coarse-grained problem (125,000 cells and message of size 400 bytes). In this case, the data scale cleanly from 1 to 512 CPUs. Furthermore, there is virtually no sensitivity to the shape of the processor grid.

Table 1 lists the CPU counts at which Sweep3D's performance was measured and the processor grid sizes used for each CPU count. CPU IDs are assigned by snaking through each dimension of BG/L's 3-D mesh network. Note that BG/L was repartitioned during the course of the Sweep3D runs; only those runs requiring more than 128 CPUs were performed on the full 512-CPU machine.

The first experiment, with data presented in Figure 14, represents a coarse-grained problem (125,000 cells and messages of size 400 bytes). In this case, the data scale cleanly from 1 to 512 CPUs. Furthermore, there is virtually no sensitivity to the shape of the processor grid.

Table 1: CPU counts and processor grids measured

CPUs	Larger dim. first	Smaller dim. first	
1	1×1	1×1	
2	2×1	1×2	
4	2×2	2×2	
8	4×2	2×4	
16	4×4	4×4	
32	8×4	4×8	
64	8×8	8×8	
96	12×8	8×12	
100	10×10	10×10	□
128	16×8	8×16	128-node partition (8×8×2)
150	15×10	10×15	512-node partition (8×8×8)
200	20×10	10×20	□
256	16×16	16×16	
300	20×15	15×20	
350	35×10	10×35	
400	20×20	20×20	
450	30×15	15×30	
500	25×20	20×25	
512	32×16	16×32	

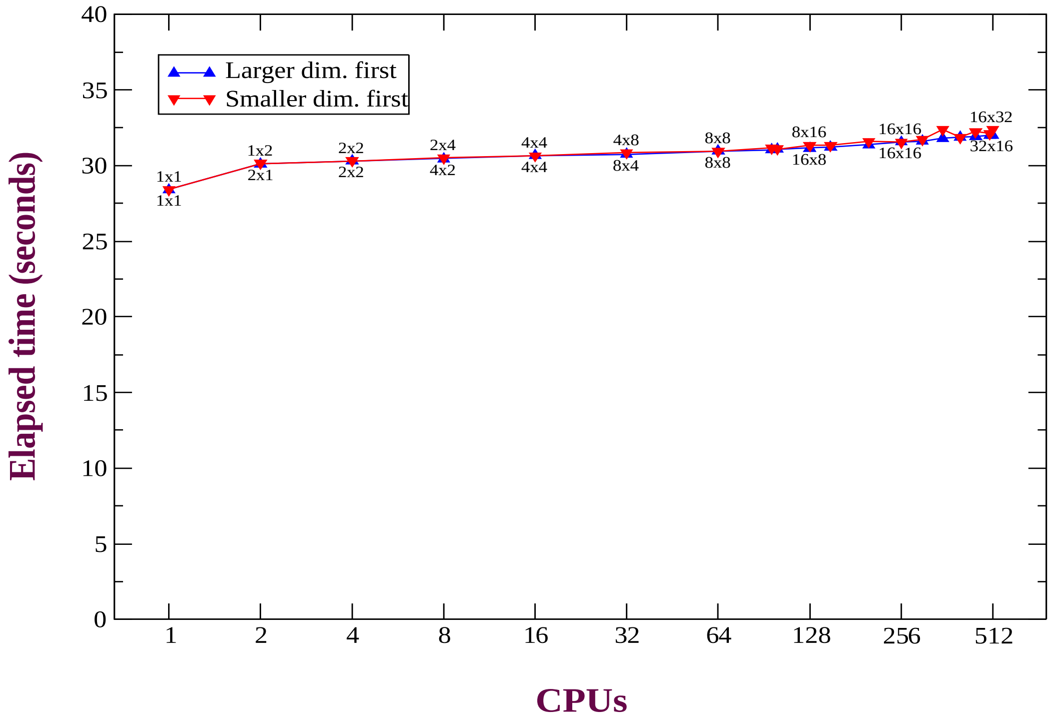

Figure 14. Sweep 3D, 50×50×50, MMI=1, MK=1.

Figure 15 represents the most coarse-grained of all of the Sweep3D runs performed on BG/L. Each CPU computes 125,000 cells and sends messages of 4000 bytes apiece. For the most part, this problem size is insensitive to the shape of the processor grid. Although performance degrades at large numbers of CPUs, this is partly caused by insufficient parallelism within the application and partly by architectural characteristics of BG/L or characteristics of the messaging layers.

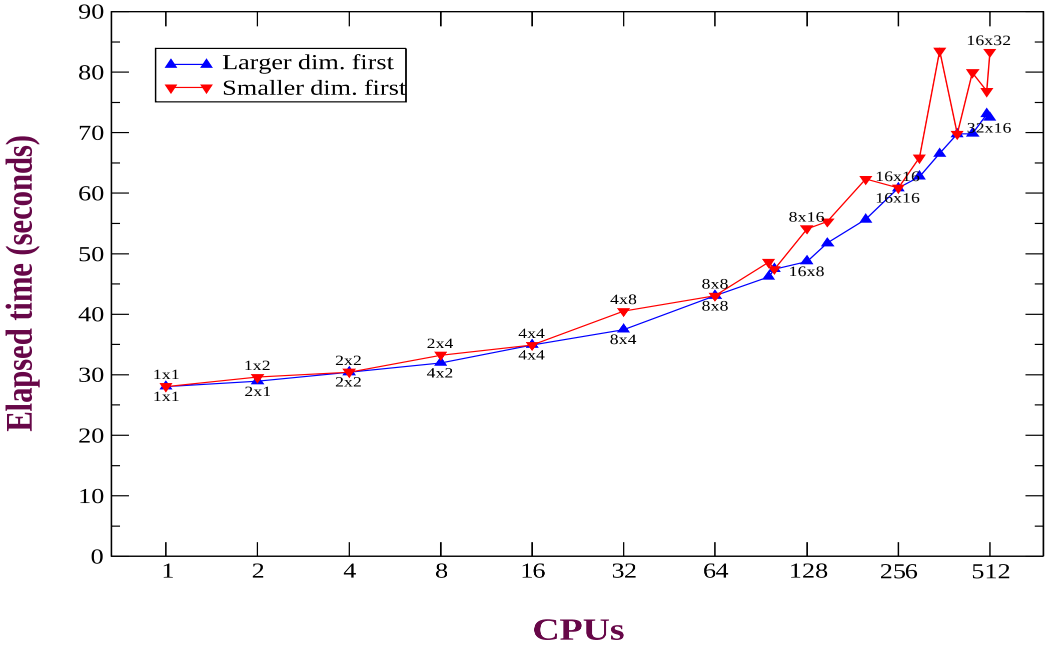

Figure 15. Sweep3D, 50×50×50, MMI=3, MK=10.

Figure 16 represents a fine-grained problem. Each CPU computes only 10,000 cells but communicates a large number of messages of 40 bytes apiece. Because of the small computation-communication ratio, these runs are highly sensitive to network performance. The rapid growth in elapsed time is caused by the heavy load on the network. Furthermore, the difference between the two curves conveys a sensitivity to the shape of the processor grid. This difference is caused partly by a lack of parallelism within the application and partly by BG/L's handling of heavy message loads.

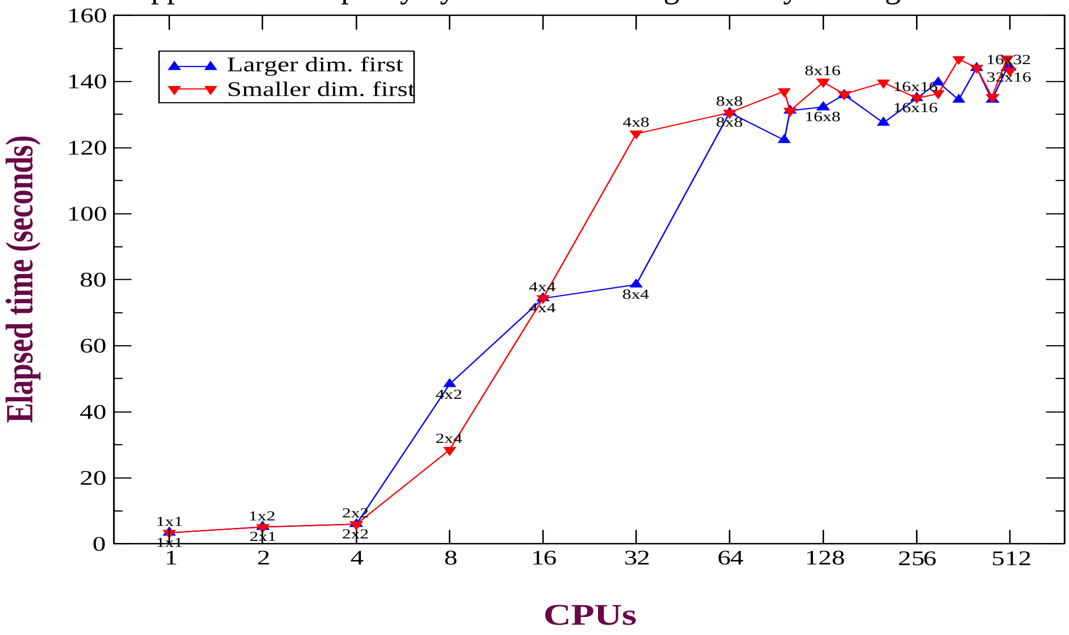

Figure 16. Sweep3D, 5×5×400, MMI=1, MK=1.

Finally, Figure 17 represents a fine-to-medium-grained problem that is quite representative of actual usage. Each CPU computes 10,000 cells and transmits a number of 400-byte messages. Up to 128 CPUs, scalability is good and there is negligible sensitivity to the shape of the processor grid. However, elapsed time increases more rapidly from 128–512 CPUs with the unusual exception of the 32×16 processor grid, probably caused by a mistake in measurement or by the job returning before the computation ended without an error message.

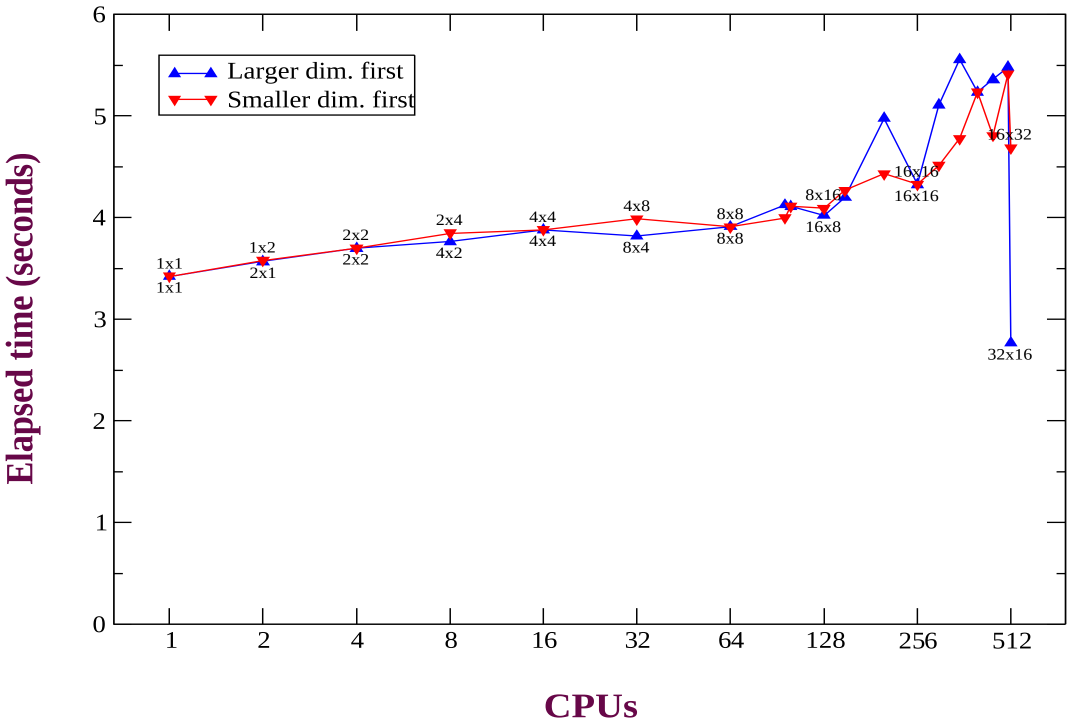

Figure 17. Sweep3D, 5×5×400, MMI=3, MK=10.

6.2 Prediction

In this section we present preliminary performance prediction number for the applications under consideration using PAL’s performance models [4], [5].

Some of the input parameters for the models are the single processor performance, which for Sweep3D was 6.75% of peak for the 5x5x400 case and 10.2% of peak for the 50x50x50 case, and the value of latency and bandwidth specified in section 4.

The blocking parameters, to which the application is very sensitive, are specified in the captions of the respective figures.

Figure 18 shows the measurements vs. the model for the coarse problem 50x50x50, in a weak scaling scenario with a subgrid of 125K cells per processor. Two blocking schemes have been employed, 10 K-planes and 3 angles per block, or 1 K-plane and 1 angle per block. We see that model predictions are exceptionally good, average error is 2.2%.

Scaling is very good out to 512 processors for the (10,3) blocking.

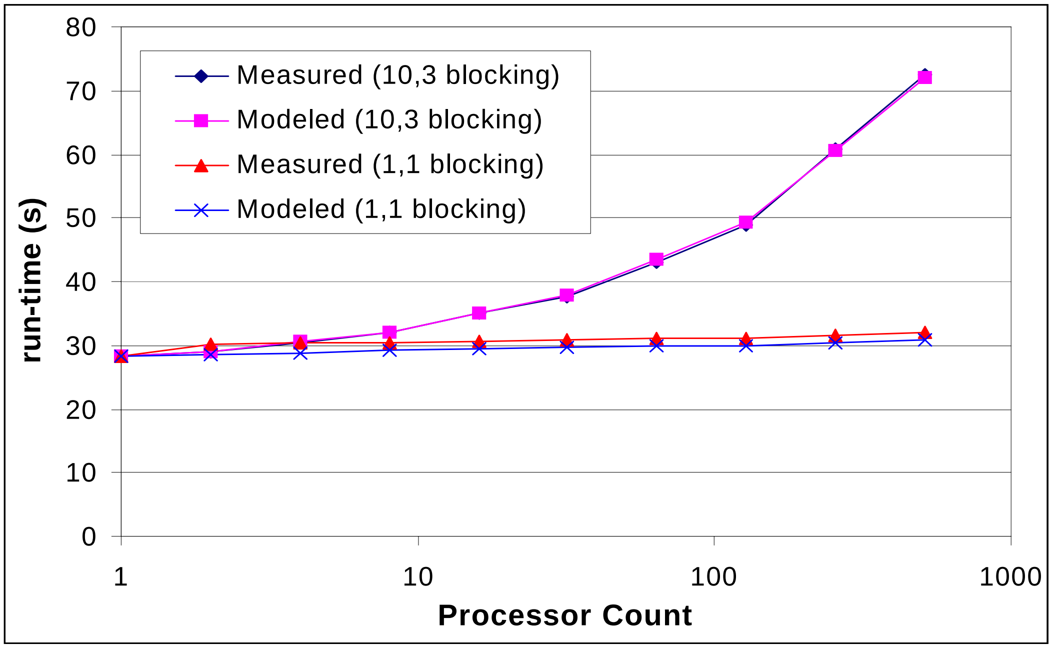

Figure 18. Performance of Sweep3D for a 50X50X50 subgrid size.

In figure 19 a 5x5x400 sub-grid per processor (10K cells per processor in weak scaling) has been utilized for one blocking scheme: 10 K-planes and 3 angles per block Similarly, the model predictions are very good, with an average error of 4.8%. Scaling is reasonable out to 512 processors.

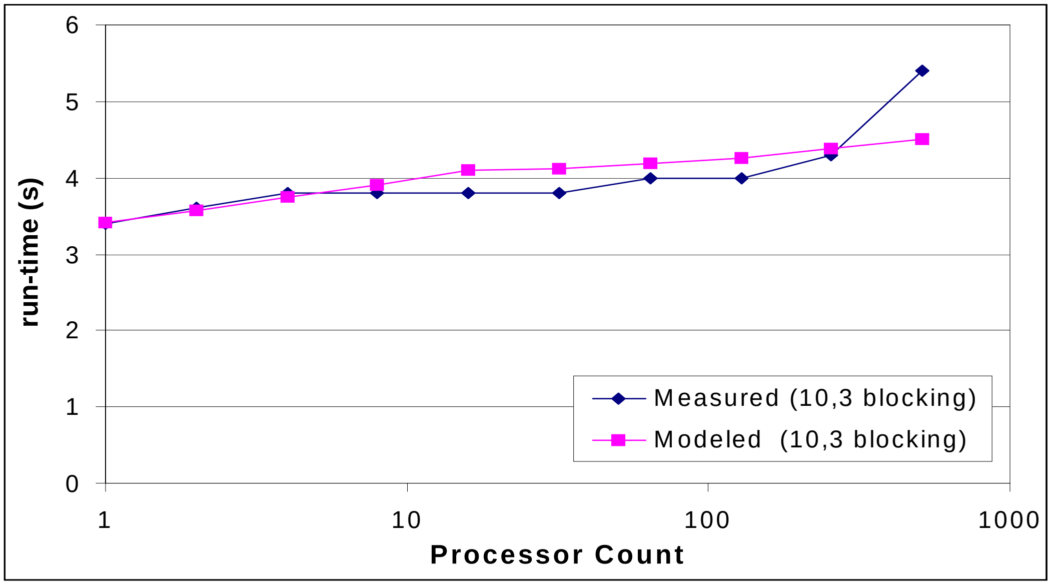

Figure 19. Performance of Sweep3D for a 10K subgrid size.

Performance predictions for Sweep3D for the 5x5x400 sub-grid per processor and 12x12x280 subgrid on the ASCI Q are shown in Figure 20.
The difference in subgrids represents difference in memory approximately between BG/L and ASCI Q.
We distinguish two regions in graph:

- up to 8,192 processors: relative performance is for an equal processor count
- above 8,192 processors, ASCI Q fixed at 8,192 and BG/L processor count increases

For the problem sizes under consideration, our models indicate that a BG/L machine of size 32K processors would achieve similar performance to ASCI Q.

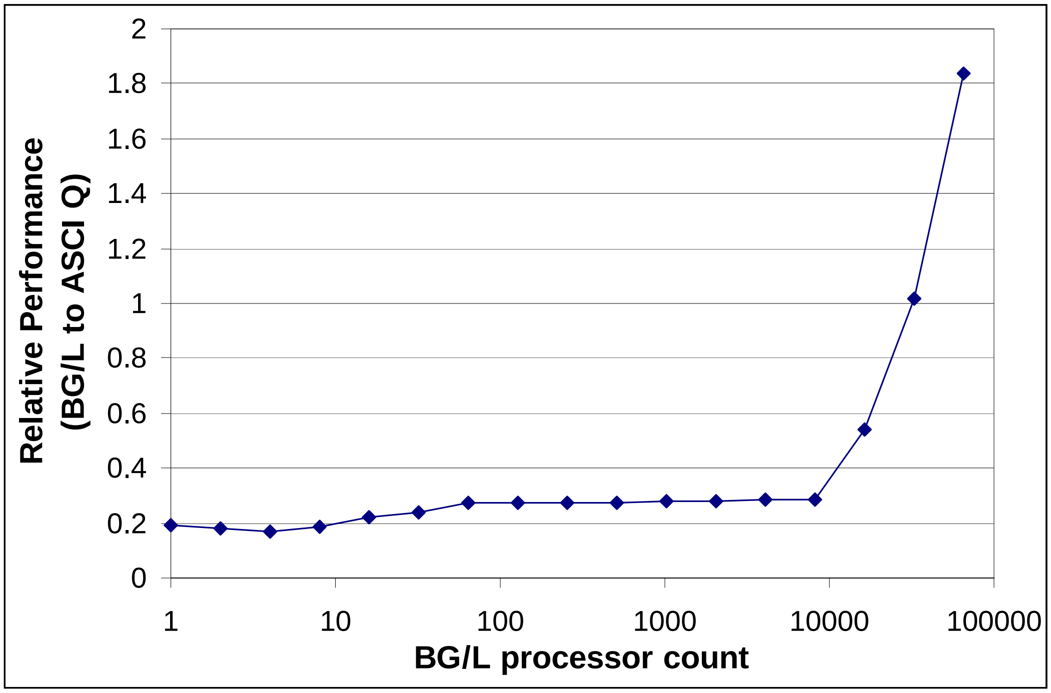

Figure 20. Relative performance of BG to ASCI Q on Sweep 3D.

Figure 21 depicts the performance of Sage, for 13,500 cells per processor (timing.input) This is a simple case for SAGE that emphasizes the communication aspects, but it is not necessarily realistic because the solver is not included. We note that measurements match the model very well, hence we used the model to predict out to 64K processors. Efficiency at 64K processors is 25% (1.32s on 1PE, 5.14s on 64K PEs).

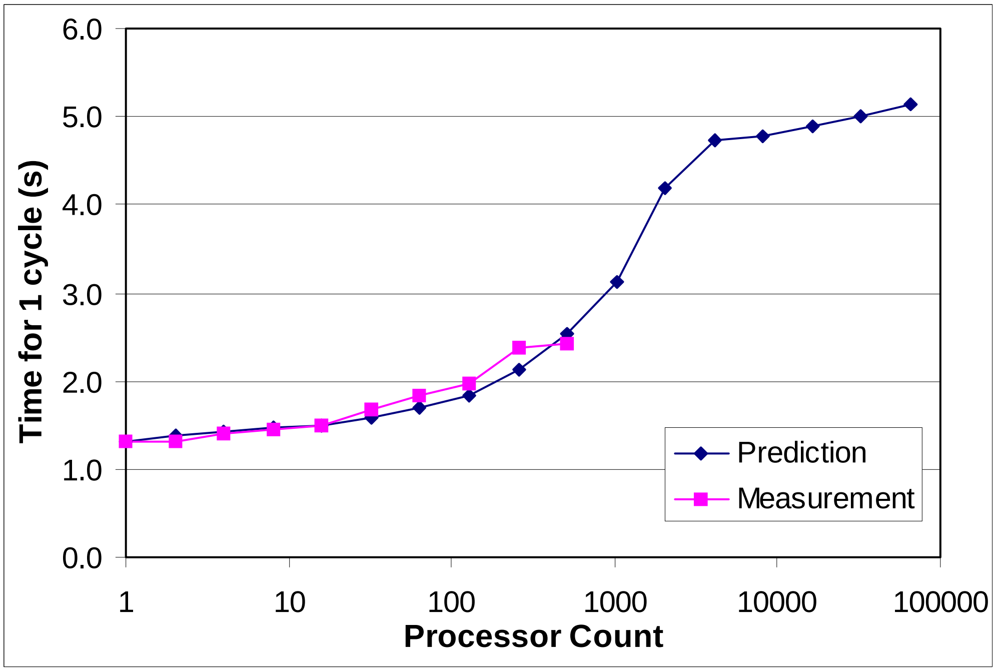

Figure 21. SAGE Performance (timing_a) on BG/L

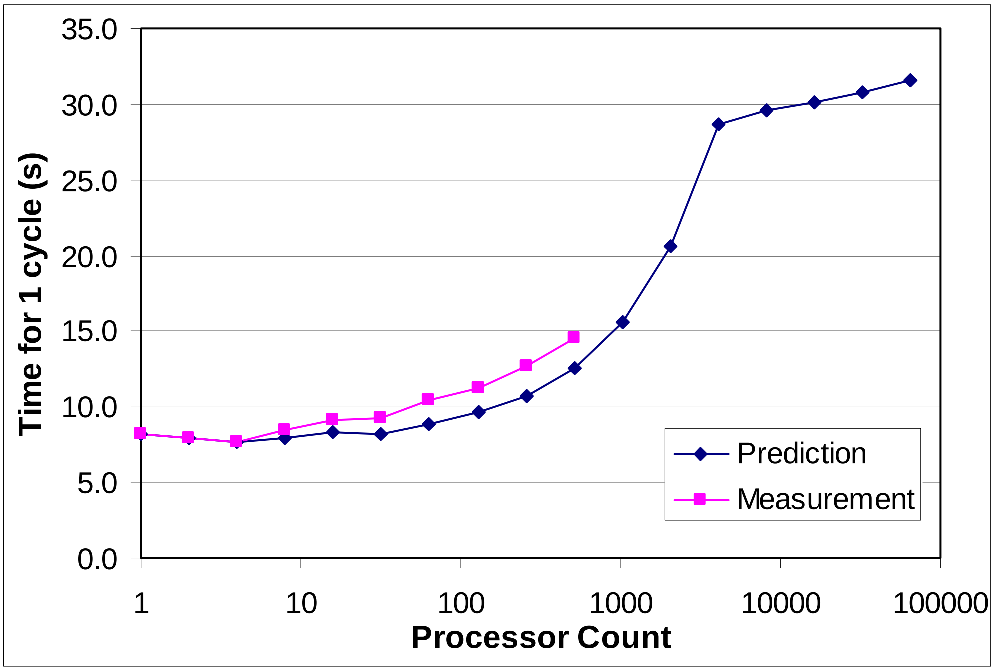

Figure 22. SAGE Performance (timing_h) on BG/L

Figure 22 shows the measured performance of SAGE v20030505 on BG/L using the timing_h. input deck. This input deck assigns approximately 35,000 cells to each processor and uses the solver on each iteration. The SAGE performance model is also shown in Figure 22 which matches the measurements to within a 10% margin. The model has also been used to predict the performance when scaling the execution of SAGE to 64K processors on BG/L. Note that the performance is expected to level off at 4K processors and above for this input deck to SAGE. This is a characteristic of the application and the position at which this plateau occurs is also a function of the system topology. For instance, this plateau occurs at 512 processors and above in ASCI Q.

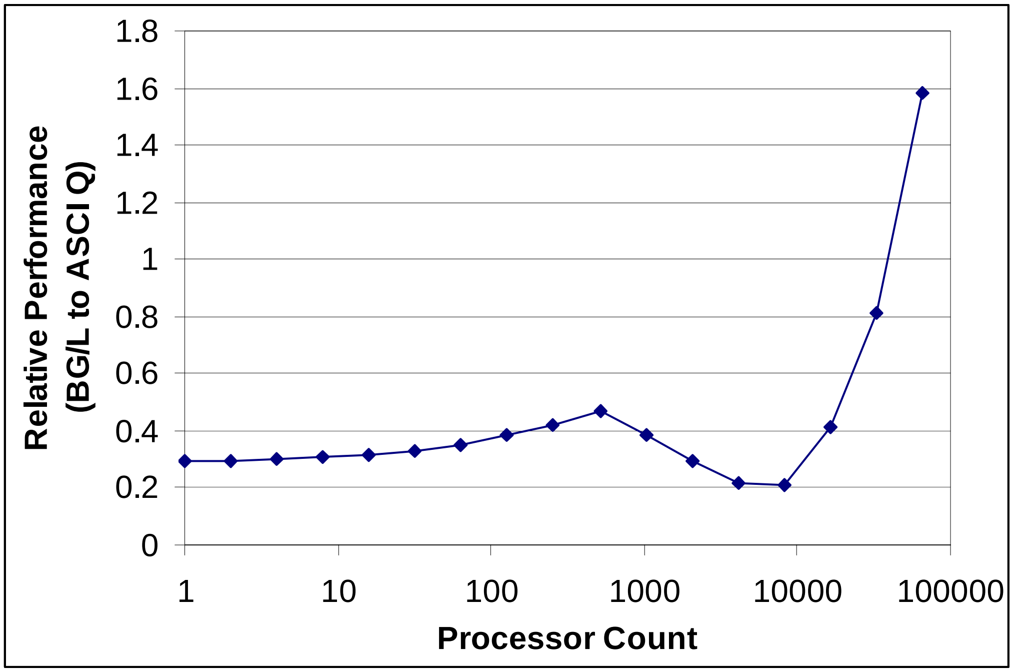

Figure 23 – Relative Performance of SAGE on BG/L to ASCI Q

Figure 23 shows the relative performance of SAGE on BG/L compared with that of ASCI Q. A BG/L processor is approximately 0.3 times the speed of an ASCI Q processor on this input deck for SAGE. However, when using over 32K processors BG/L will outperform the 8K processors of ASCI Q.

7. Conclusions

We conducted measurements and analysis, including performance prediction, on the 512-proc BG/L machine in December 2003. Given the milestone character of this configuration, special precautions have been exercised to make the data collection as accurate as possible, and the predictions as conservative as reasonable. We will update this report each time new and significant developments in hardware and/or software warrant it.

Some general comments first. Repartitioning of the machine is currently a slow process while better tools are developed for this task. A 512-processor configuration took approximately 30 minutes to load, most of the time is spent in transmitting the kernel to the nodes via the JTAG network. The IBM team is working on much faster parallel boot methods with communication via the tree network.

Virtual mode has been tried, having the advantage of doubling the number of processing threads with the theoretical advantage that better resource utilization could be achieved. While the IBM team reported some speed up in a few of the NAS benchmarks, our applications did not get a performance boost, due to the additional thread memory pressure from the additional thread.

We also noted a difference in performance based on running the same number of processors in a large partition compared to a small partition. For the purpose of our predictions we have utilized the best available data.

Various topology/configurations files provided by IBM were considered in order to improve performance through better topological placement. At the end though, the default configuration achieved the best performance for both Sweep3D and Sage.

The performance of BG/L looks promising. The network performance is good, as is the performance of the single-processor, in-line with that of other microprocessors. The scalability analysis of Sweep3D and Sage shown in Section 6. is based on conservative estimates. Obvious venues for improvement include the anticipated higher processor frequency (from 500 to 700 MHz see section 2.) and the on-going optimization work by the IBM team related to architecture-application mapping.

Predictions from 512 processors to tens of thousands of processors include a certain dose of risk, although our models are very accurate. In principle, these uncertainties could affect actual performance both ways. On the one hand a number of features that we exercised with no performance improvement and/or new ones may pan out in the future, and on the other hand scalability could be negatively affected by actual system software and hardware implementations at extreme scale.

8. Acknowledgments

We want to thanks George Almasi, Jose Castanos, Manish Gupta, Jose Moreira, Martin Ohmacht and the rest of the BG/L team at IBM Watson who actively and effectively helped us with this work.

9. Bibliography

1. An Overview of the BlueGene/L Supercomputer, NR Adiga et al, Proc. of IEEE/ACM SC2002, Baltimore, Maryland, November 16–22, 2002
2. http://www.llnl.gov/asci/platforms/bluegenel/resources.html
3. The Case of the Missing Supercomputer Performance, Achieving Optimal Performance on the 8,192 processors of ASCI Q, Fabrizio Petrini, Darren Kerbyson and Scott Pakin, Proc. of IEEE/ACM SC2003, Phoenix, AZ, November 2003.
4. Performance and Scalability Analysis of Teraflop-Scale Parallel Architectures Using Multidimensional Wavefront Applications, . Adolfy Hoisie, Olaf Lubeck, Harvey Wasserman, "The International Journal of High Performance Computing Applications, Sage Science Press, Volume 14, Number 4, Winter 2000.
5. Predictive Performance and Scalability Modeling of a Large-Scale Application, Darren J. Kerbyson, Hank J. Alme, Adolfy Hoisie, Fabrizio Petrini, Harvey J. Wasserman, and Michael Gittings, in Proc. of IEEE/ACM SC2001, Denver, November 2001.